NASA/CR-2004-212668

ViDI: Virtual Diagnostics Interface

Volume 2 — Unified File Format and Web Services as Applied to Seamless Data Transfer

Richard J. Schwartz
Swales Aerospace, Hampton, Virginia

October 2004

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results ... even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA STI Help Desk at (301) 621-0134

- Phone the NASA STI Help Desk at (301) 621-0390

- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076-1320

NASA/CR-2004-212668

ViDI: Virtual Diagnostics Interface

Volume 2 — Unified File Format and Web Services as Applied to Seamless Data Transfer

Richard J. Schwartz
Swales Aerospace, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NAS1-00135

October 2004

Acknowledgments

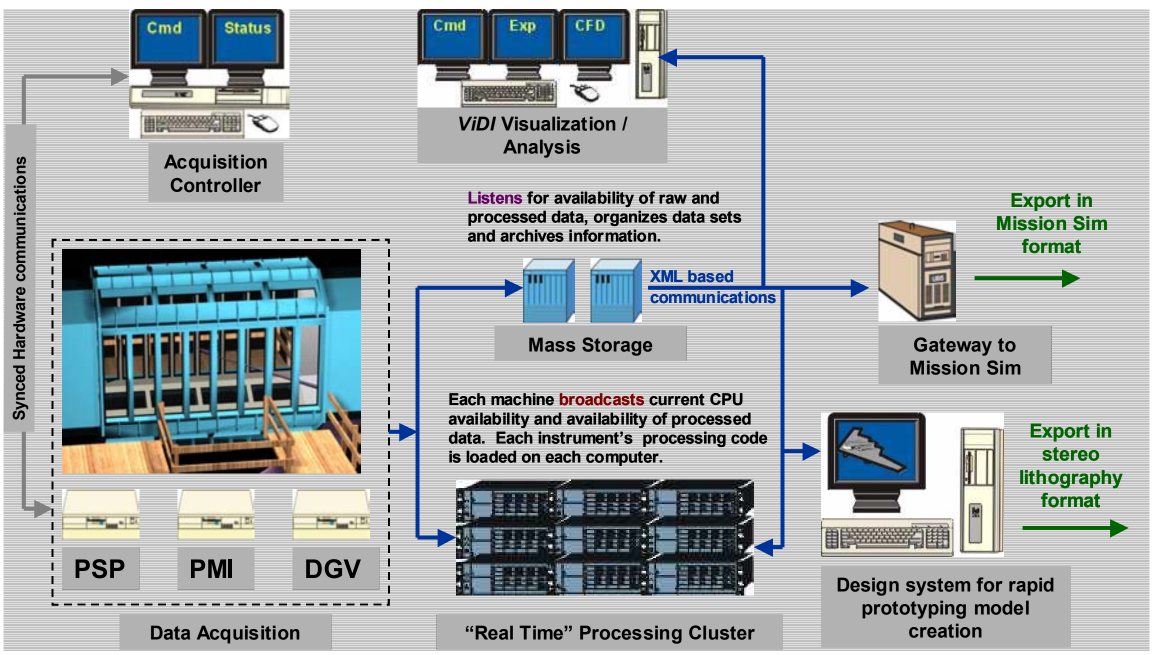
The support and vision provided by Gary A. Fleming, Dr. Jeffrey Jordan and James F. Meyers of the NASA Langley Research Center has been both essential and highly appreciated.

Available from:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 605-6000

Contents



Schematic for a fully implemented wind tunnel data acquisition system utilizing Virtual Diagnostics Interface technology.

Introduction

The desire to revolutionize the aircraft design cycle from its currently lethargic pace to a fast turn-around operation enabling the optimization of non-traditional configurations is a critical challenge facing the aeronautics industry. In response, a large scale effort is underway to not only advance the state of the art in wind tunnel testing, computational modeling, and information technology, but to unify these often disparate elements into a cohesive design resource. This paper will address Seamless Data Transfer, the critical central nervous system that will enable a wide variety of varied components to work together.

In its most elemental state, Seamless Data Transfer can be described as a carefully orchestrated flow of data. The source of the data, most likely in a "raw" state from either experiment or computation will be processed at intermediate stations for delivery to its destination in an accurate and usable form. With numerous instrumentation systems, computational resources, analysis packages and custom data delivery requirements to a diverse set of customers, the handling of these large data sets mandates an architecture designed for flexibility and expandability. Thus, two challenges have to be met:

1. The architecture must be able to handle scientific data sets,
2. It must be able to route the information from place to place as efficiently as possible.

In the search for solutions, a desire to use commercial off the shelf items as much as possible has been critical, as well as the need to develop a system based on technologies that will endure without rapid obsolescence. This paper will address the first challenge in the *Unified File Format* section 1, while the second challenge will be discussed in the *Designing Seamless Data Transfer* section.

Unified File Format

Format Description

The term Unified File Format (UFF) refers to a type of computer file that can contain many different types of data from a variety of sources. In this context, UFF refers to a file format that has a large *scope*. The scope of a data file is defined as the range of data types that are to be included. This can vary from the very simple, such as a list of integer numbers, to the very complex, which might include a variety of data types (binary, integer, floating point, images, image palettes). The goal of choosing a file format is to ensure encapsulation of the data scope from the onset.

Traditionally, one type of data has been stored per file type. Images populate TIF, BMP or JPEG files, numerical arrays in text format are in TXT files, and drawings may be found in the IGES format. The Unified File Format looks to do away with such a compartmentalized convention. A block of data, such as a drawing or an image may remain in a certain format (IGES, JPEG), but that block of data will be incorporated into the Unified File Format. In the UFF, data will be organized in separate, well-organized structures, as shown in *Figure 1*. This is similar in concept to a hard drive organizational structure, with a root, directories, and subdirectories. Each part of the data set, be it an image, a drawing, or an array of numbers representing raw or processed data, will be located in a specified location of the file. It is also possible that particular data may be held in an ancillary file while the main data file contains a pointer, or a *hyperlink,* to the data. Currently, this is how HTML incorporates images. The image data is held separate from the

HTML file and accessed by an application reading the HTML files. Thus, a file format that allows data to be organized in a directory structure, and one that includes hyperlinking is required.

Two file formats have emerged as possibilities. The first, HDF 5 (Hierarchical Data Format) is the latest offering in this series of file formats from the National Center for Supercomputing Applications (NCSA). HDF 5 (which is not compatible with the previous HDF 4 version due to the radical advancements contained within) was envisioned from the start as a file format for large (gigabyte and beyond) data sets. This paper will limit itself to the discussion of HDF5, and all references to HDF will be assumed to be referring to version 5.

The other format is the eXtended Markup Language, or XML. XML is currently favored in the IT community, spreading its presence across all fronts, from web applications to the latest distributed computing projects under the Microsoft "dot net" banner. The remaining portion of this section will look at the benefits and shortfalls of both file formats.

File Format Attributes: Self-Describing

Almost all computer files possess *header* information along with whatever data they may contain. This header information is keyed to a specific purpose, and generally a specific computer program. For example, the actual text of a word processor file is only one component. Header information and tags identify bold or italicized characters, font sizes and styles, and other attributes to properly re-create the document. Typically, each computer application will have its own distinct header structure and tags. However, there is a newer style of data file called a *self-descriptive* data file. When such a file is opened in an appropriate viewer, the viewer is able to reconfigure itself based on the information contained within the file header. For example, if the header specifies a set of integers in binary format, the appropriate tools would be brought forth to view and manipulate that data. Likewise, if a set of characters making up a paragraph (or a whole document) were found, a word processor-like set of tools would be presented to the user. Most importantly, such file formats allow users to specify their own types of data (for instance 128 bit unsigned integers, *see Figure 2*). Additionally, the file may contain hyperlinks, which the viewing application will interpret as pointers to external files. These links provide access to data or executable code either locally or over a network.

HDF and XML are both self-describing. All the information needed to describe the data the file contains is stored within the file itself in a predefined manner. This is accomplished through the use of labels in HDF and tags in XML. These labels and tags act as mini-headers that describe attributes of the data. Both data formats allow links to other files (local or over a network), and as described above, the data is organized in a directory tree structure. HDF files typically contain all of the header properties within the HDF file itself, while XML files can contain the data with the main data file, or rely on other files to provide the header information.

File Format Attributes: Binary or Text

While all computer information is ultimately stored as binary data in memory or storage, the way a file handles data can vary. A file is said to be "binary" if the data held within it is optimized in a way that allows each bit of information to be transformed into numerical values most closely orientated for computer processing and storage. Without the appropriate translation, the contents of a binary file appear to be meaningless. However, binary files are compact, and computers handle them very quickly. Most computer applications use binary file formats.

In contrast to binary, data can also be stored in an ASCII text format. Here, each value of data is given a character (letter or number). Each character will take up 8 bits, and a single number, such as fifty-five, will occupy two 8 bit sections of memory, while five thousand will occupy four 8 bit memory sections. In binary, fifty-five would require only one 8-bit memory section, while five thousand would require two. Thus, ASCII text data can require far more storage space for the same information. The advantage of ASCII text is in the readability of the file. Any simple text editor can open an ASCII text file and at least gain some insight into the file contents. Modifications can easily be typed in.

HDF uses a binary format, while XML is a text-based format. In order to read and write a HDF file, a set of library functions must be called in application software. This requires a special program to be written to write and extract the data from HDF files. A XML file can be opened and modified in any standard text editor.

Since HDF is designed to handle large scientific data sets, the available library functions a programmer would use are optimized for the types of data used with Seamless Data Transfer. A HDF file with a given set of data will be markedly smaller than the corresponding XML file, thus making storage and transfers more manageable.

If one is adamant about taking the text-based route, it is possible to use an intermediate translation to convert binary into a "binarized" ASCII, which, while in ASCII format, would be more compact. It would not be readable without the proper "de-binarizing".

When a text-based file has to use binary data, this data is most often *hyperlinked* to the text-based file. As data is transported from machine to machine, and from one network to another, hyperlinks often need to be updated to ensure they point to the correct files. Thus, the more binary files that are hyperlinked to a text file, the more opportunities one has to have "broken" hyperlinks that point to non-existent file addresses. This is akin to loosing the data, as a substantial effort may be involved with fixing the hyperlinks and revalidating the integrity of the entire data set.

File Format Attributes: Compression

When dealing with gigabytes to terabytes worth of data, data compression becomes a critical issue. Without the proper compression schemes, either the size of the data files can overwhelm even the most modern computer systems, or degradation of data due to compression can render it useless.

There are two types of data compression. The first is called *lossy*. This method, often used in the JPEG, MPEG and MP3 formats, trades away data for mathematical information that will allow the computer to recreate a close approximation. Such methods are often based on exploiting the limits of human vision or hearing. Lossy schemes are almost universally unsuitable for scientific data sets. The other type of compression is *lossless*. Lossless schemes compress data, but often not to the extent that a lossy scheme would. However, this is the only way to ensure the integrity of the scientific data.

HDF incorporates a lossless compression scheme known as GZIP. More schemes may be developed in the future. An important, and unique feature of HDF is that the data can be *chunked*. Chunking allows certain portions of the data to be grouped together in "chunks". Each chunk can be compressed and decompressed individually, so a user can retrieve a half-megabyte of data out of a multi-gigabyte file without the need to decompress the whole file.

XML does not support any form of compression. A work around would be to encode binary data, as mentioned in the previous section, use a separate data compression tool, and then include the compressed, encoded data in the XML file. Again, this would be illegible to a text editor, and would require both decoding and decompression.

File Format Attributes: Development

In choosing a file format, consideration must be given to the ease or difficulty of developing software that uses the format. Both HDF and XML are well endowed with development tools.

Since HDF is stored in a binary format, access to the data must be though specific code designed to read and write HDF documents. The NCSA provides developers with an API (Application Programming Interface, a set of documented libraries) to manipulate HDF files. The API works with many of the leading programming languages such as C++, Java, and Fortran. The NCSA provides detailed documentation of the API with examples to get developers started.

Though XML is text based, developers can obtain API's to create and maintain XML files. Document Object Model (DOM) and Simple API for XML (SAX or SAX2) are two of the leading API's for XML. These API's allow developers to program XML documents with anything from scripting languages to more complex languages such as C++ and Java.

Designing Seamless Data Transfer

Web Services: Philosophy and Examples

Web services are critical to the handling of data over a large network. This is a complex, emerging technology that will lead to greatly enhanced Internet functionality in the future. The core of web services is XML data handling.

A large network utilizing web services will differ from the current network architecture in some fundamental ways. Today, machines have to target requests over a network to other individual machines. A user looking for information must know the address of the computer with the proper information. A web search can be done, but the majority of the search results are often undesired. Search engines build tables based on the content in the HTML web pages and store the data for later rapid retrieval. In contrast, web services would allow computers to listen for requests over the network, and broadcast out the pertinent information to the correct address. As an example, imagine a person's desire to complete a morning meeting in New York, lunch in Paris, and spend a quiet evening in his Mediterranean beach villa. Today, barring a good travel agent, the user would have to map out each section of the travel, contact his favorite restaurant in Paris, and have his valet alerted in the coast side home. Using web services, the scenario would go through the following process:

The user would tell his handheld computer/phone/GPS unit that he wanted to have lunch at 1:00PM (GMT +2 hours considering daylight savings time in Paris). The computer would contact the restaurant in Paris in a fashion analogous to today's Internet. Next, the user's computer would ascertain from the GPS that Paris is 3000 miles away, and that only a supersonic flight would allow the user to meet his schedule. The computer would now access web services and go out on the net to seek computers that are broadcasting airline ticket purchases. After finding several airline computers selling tickets, the computer determines that only two offer supersonic service. The computer would book the best flight to meet the schedule. Next, it would seek out computers broadcasting charter private jet flights from Paris to the closest city to the Mediterranean villa, and repeat the procedure.

What is key here is that each airline is broadcasting that they provide a basic service, and then more detailed queries and data processing can be made. Following automatic procedures, tickets are purchased, and electronic funds are passed automatically from the bank computers. As envisioned by today's industry leaders in IT, these services will utilize tools that process XML files. What is also important is that the processing power required to conduct this transaction is distributed. With this example transaction, the computations (which require solving the "traveling salesman problem", comparative cost analysis, and extensive network communications) may be carried out on the users hand held device, the airline or bank computers, or some arbitrary third party computer. The required resources are allocated on the fly between machines that broadcast they have available processing and bandwidth capabilities. Numerous CPUs may ultimately produce the end result. All of this will be invisible to the user, who will ultimately have his transaction processed in a matter of a few moments.

Switching to a wind tunnel experiment that employs a web service based network, the data acquisition and processing might go as such:

A user will make a request that data be taken at the current test conditions, as depicted in *Figure 3*. Multiple instrument systems (Pressure Sensitive Paint, Projection Moiré Interferometry, Doppler Global Velocimetry, etc.) each are triggered in the proper sequence. Megabytes of raw data are collected in a few seconds. The set of computers that just acquired the data now go out onto the net and listen for broadcasts from machines that are ready to process the data. The acquisition computers will send some or all of its raw data to the processing machines, *Figure 4*. These machines will broadcast their utilization rate, so additional data may be routed to less utilized network components. A mass storage device will also broadcast that it is collecting all of the raw and processed data for archival purposes. A copy of the data will be sent there. As the processing proceeds, a Virtual Diagnostics Interface (*ViDI*) computer will broadcast a request for data taken at a certain wind tunnel test condition. This data may be routed from the processing computers as soon as the data is ready, or it may be retrieved from the mass storage unit if it was processed in the past. In addition, computational results for the matching condition will be sent to the *ViDI* computer. Then users may analyze the results. Another computer will broadcast a request for data required by the Mission Simulation group, who can determine the utility of a given airframe design based on the output of the wind tunnel performance data. This machine will gather the information required and translate it into a format the mission simulation group can use. They can recommend changes, which will make its way back to the rapid prototyping systems for the next iteration in the design cycle. This process can be defined in the time span of hours or days, as opposed to the current weeks to months. In this way, data is transported to the correct user at the correct time with a minimum of human intervention. This is seamless data transfer.

Choosing a File Format

Seamless Data Transfer relies on a Unified File Format and the ability to automate complex data transfer, processing and interpretation tasks. This paper has looked at how to begin implementing these technologies, and has asked which file format, HDF or XML, would be better for the task. *The answer is to use both*. Each format has unique and powerful attributes, while each also lacks key features present in the other.

The scientific data will be held in HDF. This is the best way to maintain the disparate data, compress the information, and organize it in a logical fashion. Using HDF also eliminates the need for extensive hyperlinking of data sets.

Another powerful feature of HDF is the ability to contain palettes for images as an integral part of the file format. Common palettes for images are crucial, as *the colors are the data.* To make a direct comparison between experiment and computation, and to correlate between maximums and minima between differing measurements, the use of a common palette can be critical. A single palette in an HDF file can be made available to all data sets, and a change to one palette can be distributed to all of the visualized data if desired.

As envisioned, each element of the test (instrumentation, computational results, *ViDI,* tunnel Data Acquisition System) would be responsible for outputting calibration information and raw and processed data. This data would be saved in a HDF file that would strictly follow a given template. In one scenario, this file would be held in a mass storage area, and catalogued in a database for rapid retrieval. Upon a request for data, the desired information would be rendered into a new HDF file. Thus, if the user requests only processed data from computations and multiple instruments for a given test point, that data will be placed within a single HDF file, then routed to the end user. A second scenario would directly sub-sample the data from a test and forward it to the analysis computer for real time monitoring.

The problems of transporting the data from place to place in a timely, organized and automated way cannot be handled by HDF. For this, web services based on XML are ideal. The HDF files will be hyperlinked to the XML files. Since each HDF file can contain a variety of different data in one package, this will minimize the amount of hyperlinks required. The various elements on the network (instruments, mass storage, visualization tools, etc.) will interact using the web services in order to route the data to the correct locations and optimize the resources of the entire network. This will allow both real time analysis and detailed post-test investigation of the test data.

The concept of using HDF with XML is also being explored by the NCSA. NCSA has begun this process by looking at three distinct areas of implementing the two technologies together. The first is a Document Type Definition (or an XML Schema) that allows an XML file to interact with a HDF file. Uses for this include[1]

- Viewing structure and contents of HDF5 file using a Web browser
- XML as a catalog record for locating datasets
- XML as an intermediate form for programs
- Generation, validation, and reconstruction of HDF5 files
- XML as intermediate to other formal languages and file formats
- Store XML in archive or in dataset as machine readable documentation
- Templates, skeleton files, etc.

A second project undertaken by NCSA is an addition to a utility they call *h5dump.* This utility will output HDF data in a "human readable" form. It has recently added output support for XML. And lastly, a Java tool, *h5gen,* can read an XML description of an HDF file, and then create an HDF file from the XML information.

Additional work with XML and scientific data sets has been conducted at the NASA Goddard Space Flight Center. An XML schema to support scientific data sets called XDF (eXtensible

Data Format) has been developed for use with astronomical data. In a white paper[2] describing the use of XDF, the question of large data sets and binary data are discussed.

> "There is an ongoing debate about how best to use XML when faced with large data files. First of all, most existing large data files are in binary format and one simply can not mark these up with XML tags."

The author continues to explain that a separate compression scheme, call XMIL, has been written to allow for compression and storage of binary data, as was alluded to in a previous section of this paper. Ultimately, however, the author of the white paper on XDF concludes;

> "It is because of these problems with large data files and because of the huge legacy of data files in binary and ascii format that are not marked up, that XDF is careful to allow both varieties of data to be included. One can fuly mark-up data and leave out formatting information or one can use old fashioned fixed width formats for records or one can express the binary formats used."
> *(Grammatical and spelling errors retained from the original document)

The shortcomings attributed to XDF (which is probably well suited for the data handling tasks it was designed for) are addressed in the Seamless Data Transfer scheme by the implementation of HDF.

File Format Viability

When a file standard is chosen at the start of such a large, long-term project, the viability of the standard has to be examined. Apart from the technical issues discussed above, the likelihood that the format will be supported for the long term is critical.

Industry wide, HDF5 fills a small, specialized segment of computer user needs. Several major players currently use the HDF5 format, including the NASA- EOS project and the Department Of Energy's Advanced Simulation and Computing Program. In addition, the NCSA has been recognized for its work with HDF5 by winning a 2002 R&D 100 award from R&D magazine. Based on history, performance, and need, it appears that HDF5 will be a long-term technology.

The concepts behind XML have had a long gestation period. XML is a subset of SGML[3], an ANSI standard markup language developed in the late 1970's and early 1980's. However, it has not been until the maturation of the Internet and the need to go beyond an HTML type of file that the broad computing infrastructure made XML appealing. As of late 2002, numerous companies (Microsoft, Oracle, IBM) are all introducing XML based features and extensions. A review of the XML.com web page gives a good idea of the vast amount of XML work as well as the controversies surrounding competitive technologies to implement XML. Every indication is that web services based on XML are just starting a long and important cycle in the continuing development of information technologies.

Costs Comparison

The costs associated with developing Seamless Data Transfer are well beyond the scope of this paper. However, the initial costs for implementing HDF and XML can be discussed. At the base level, both are free.

All HDF material developed by the NCSA is freely distributed. This includes the API, utilities, and documentation. While third party viewers for HDF files exist, it is currently envisioned that most of the software developed for Seamless Data Transfer will focus on the NCSA material.

The basic XML is also free, as it is simply a text file. XML browsers (such as a beta of Mozilla[4]) are available for download. However, the details of implementing web services for Seamless Data Transfer have to be further defined before individual products or development suites can be recommended.

Software for Scientific Data Analysis

One aspect of Seamless Data Transfer pertains to making data available to anyone who may rightfully gain access to the information. This includes researchers beyond access to the formal hardware and software dedicated to Seamless Data Transfer. Such users will most likely use a number of commercial software packages that are dedicated to scientific data analysis and visualization. Thus, a file format that enables them to easily access the data is important. IDL by Research Systems supports HDF 5 in its latest version. Two additional packages, PVWave by Visual Numerics, and MatLab by The MathWorks currently support HDF 4 (which is not compatible with HDF 5). However, in discussions with the technical support personnel from each of these companies (October 2002), information was received that suggested HDF5 support was under development. Both companies confided that there have been a number of requests for HDF5 support for their software.

Conclusions

At the core of Seamless Data Transfer is the manipulation and transfer of large scientific data sets. A survey of existing technologies has shown that the best way to handle such data sets is to use the HDF 5 file format, expressly developed for such use by the NCSA. The network transport required by such a system will best be served by the emerging web services, technologies based on the XML standard, By using a combination of HDF and XML, both data handling efficiency and the use of COTS software can be maximized.

Summary

The table below summarizes the key concepts identified as issues concerning the implementation of the Unified File Format with Seamless Data Transfer.

Property	HDF	XML	Comments
Self Descriptive	X	X	Both require a compatible viewer
Binary data	X		Binary format critical to large scientific data sets
Multiple data types	X		Any type HDF, only text on XML
Compression	X		Limited to GZIP, chunking is a powerful feature
Imbedded Image Support	X		HDF supports images and palettes XML requires hyperlinks to images
Development API	X	X	Both formats have resources to aid development
COTS scientific visualization	X		HDF 5 is being implemented in leading COTS scientific analysis software
Web Services		X	Critical for advanced networked operations
Standardization		X	Being adopted widely by developers
Free API	X		HDF API free, many commercial XML APIs
Viability	X	X	Long term prognosis for both formats is good.

Table 1. Summary of features provide by each file format.

References

1. A DTD for HDF5: Design Notes by Robert E. McGrath, NCSA HDF web page
   http://hdf.ncsa.uiuc.edu/HDF5/XML/dtd-report.html

2. XDF, the eXtensible Data Format for Scientific Data. Ed Shaya, ADC/NASA/RITSS
   http://xml.gsfc.nasa.gov/XDF/XDFwhite.txt

3. Learning XML by Erik T. Ray
   O'Reilly & Associates, Inc., 2001. ISBN 0-596-00046-4

4. Mozilla Browser with XML support, by the Mozilla Organization
   http://www.mozilla.org/projects/seamonkey/release-notes/m18.html

Bibliography:

The National Center for Supercomputing Applications home page for HDF
http://hdf.ncsa.uiuc.edu/

O'Reilly XML.COM, a service to XML programmers on the web
http://www.xml.com/

Understanding Web Services, XML, WSDL, SOAP and UDDI by Eric Newcomber
Addison-Wesley, 2002. ISBN 0-201-75081-3

Professional XML for .Net Developers, by Dinar Dalvi et al.
Wrox Press Ltd., 2001. ISBN 1-861005-31-8

HDF4 and HDF5 Performance Preliminary Results, Elena Pourmal, IV HDF-EOS Workshop, September 19-21, 2000.
http://hdf.ncsa.uiuc.edu/HDF5/HDF4vsHDF5/Performance_talk/

Introduction to HDF5, NCSA/University of Illinois at Urbana-Champaign, May 2002
http://hdf.ncsa.uiuc.edu/HDF5/papers/HDF5_overview/index.htm

eXtensible Data Format (XDF) web page
http://xml.gsfc.nasa.gov/XDF/XDF_home.html

Research Systems IDL Data Analysis web page
http://www.rsinc.com/idl/

Visual Numerics PV-Wave web page
http://www.vni.com/products/wave/specs.html

Figures

Implementing UFF

The UFF file will be a self descriptive file with a directory tree structure, as pictured to the side.

Each individual instrument would have the common area filled, and only its own data directories filled. A separate "Data Compiler" can bring together the different data sets and create a unified UFF file containing all of the desired data. This will ease transport of massive data sets, as each unified UFF file may be gigabytes in size.

```
Root \
   Test Note
   Run Note
   Point Note

   Common Parameters
              Velocity
              Temperature
              Pressure
              :
              :
  ViDI
              ViDI  Notes
              3D Facility model
              Instrumentation layout
              Simulated camera views
              :
              :
  PSP Parameters
              PSP Notes
              Calibration information
              Background Images
              Raw data Images
              Processed Images
              :
              :
 PMI Parameters
              PMI Notes
              Calibration information
              Background Images
              Raw data Images
              Processed Images
              :
              :
  CFD
              Mesh Model
              Solution file
              :
              :
```

Test Note - constant for all files for a test.

Run Note - constant for each file in a run.

Point Note would vary from each file.

Common Parameters would hold data that defines the tunnel conditions, and other information that would eliminate the need for duplicating the same data in multiple locations.

A series of directories and subdirectories, organized in a like manner will be implemented for each technique. A given set of logical rules for each instrument and computational method will be developed to help define the file format in an efficient, predictable, and expandable way.

Figure 1. Organizational structure for the unified file format

HDF Data Types

HDF supports a wide variety of data types natively.

Additional data types can be created, such as a 128 byte integer, if desired.

In addition, image data and associated palettes are an integral part of the HDF file format.

HDF native data type support

Example	Corresponding C Type
H5T_NATIVE_CHAR	char
H5T_NATIVE_SCHAR	signed char
H5T_NATIVE_UCHAR	unsigned char
H5T_NATIVE_SHORT	short
H5T_NATIVE_USHORT	unsigned short
H5T_NATIVE_INT	int
H5T_NATIVE_UINT	unsigned int
H5T_NATIVE_LONG	long
H5T_NATIVE_ULONG	unsigned long
H5T_NATIVE_LLONG	long long
H5T_NATIVE_ULLONG	unsigned long long
H5T_NATIVE_FLOAT	float
H5T_NATIVE_DOUBLE	double
H5T_NATIVE_LDOUBLE	long double
H5T_NATIVE_HSIZE	hsize_t
H5T_NATIVE_HSSIZE	hssize_t
H5T_NATIVE_HERR	herr_t
H5T_NATIVE_HBOOL	hbool_t

Figure 2. Native HDF Data types

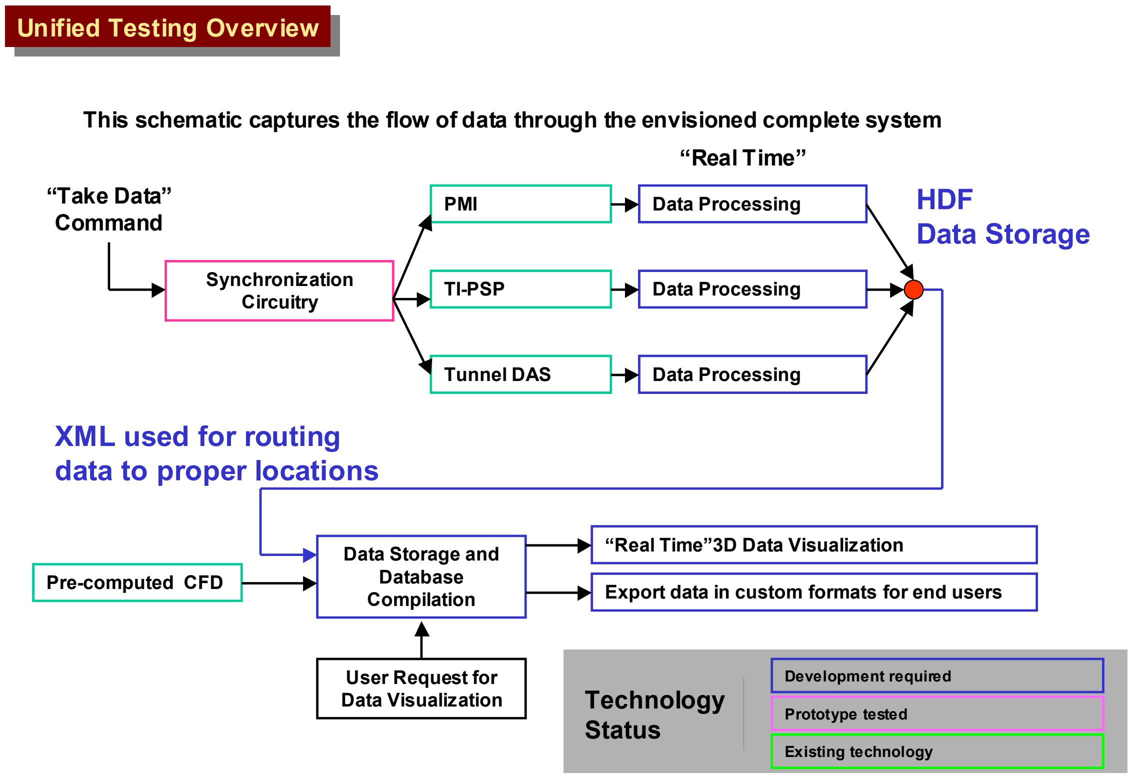


Figure 3. Pathway for data in wind tunnel test scenario

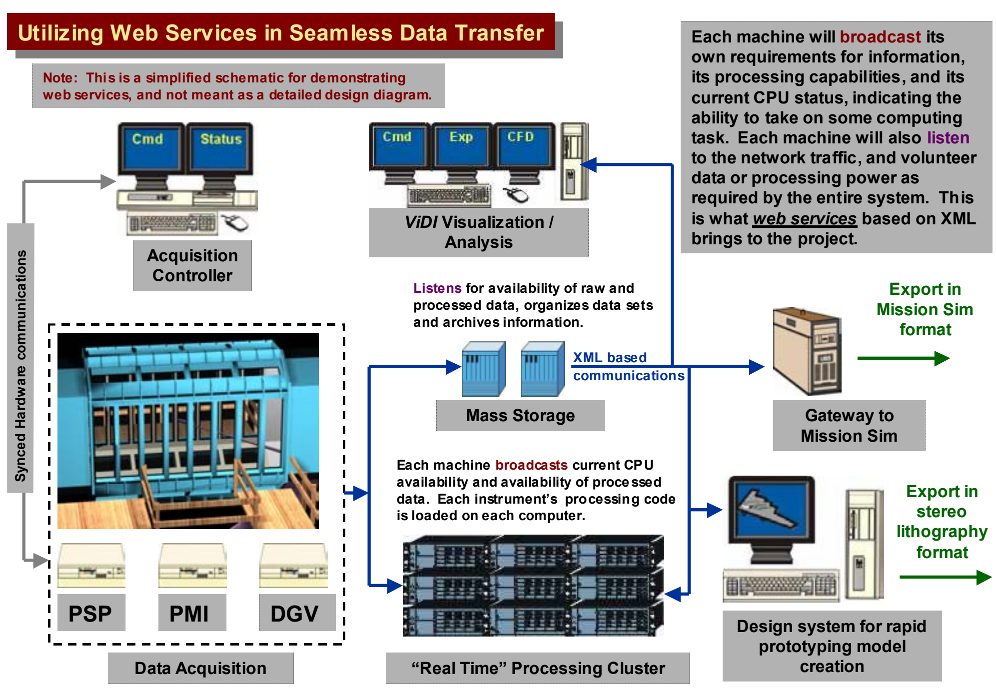


Figure 4. Generalized schematic describing web services for Seamless Data Transfer

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01- 10 - 2004	Contractor Report	

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
ViDI: Virtual Diagnostics Interface *Volume 2 — Unified File Format and Web Services as Applied to Seamless Data Transfer*	NAS1-00135
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
Schwartz, Richard J.	5e. TASK NUMBER 25RBG
	5f. WORK UNIT NUMBER 23-706-21-11

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
NASA Langley Research Center Hampton, VA 23681-2199	

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
National Aeronautics and Space Administration Washington, DC 20546-0001	NASA
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) NASA/CR-2004-212668

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 35
Availability: NASA CASI (301) 621-0390 Distribution: Nonstandard

13. SUPPLEMENTARY NOTES
Langley Technical Monitor: Gary A. Fleming
An electronic version can be found at http://techreports.larc.nasa.gov/ltrs/ or http://ntrs.nasa.gov

14. ABSTRACT
The desire to revolutionize the aircraft design cycle from its currently lethargic pace to a fast turn-around operation enabling the optimization of non-traditional configurations is a critical challenge facing the aeronautics industry. In response, a large scale effort is underway to not only advance the state of the art in wind tunnel testing, computational modeling, and information technology, but to unify these often disparate elements into a cohesive design resource. This paper will address Seamless Data Transfer, the critical central nervous system that will enable a wide variety of varied components to work together.

15. SUBJECT TERMS
Simulation; Virtual diagnostics; Data files; HDF; File format; Global instrumentation; Wind tunnel testing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email: help@sti.nasa.gov)
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	19	(301) 621-0390